

## INTRODUCTION

Cet exercice simule une administration : des bureaucrates avec un grade, des formulaires spécialisés, et un stagiaire capable de créer ces formulaires à partir d'un simple nom. L'objectif est d'expliquer clairement le fonctionnement du code, la hiérarchie, la création dynamique de formulaires et l'interprétation complète de la sortie du programme.

### 1. ARCHITECTURE GLOBALE

- Bureaucrat : un employé avec un nom et un grade (1 = plus haut, 150 = plus bas). Il peut signer et exécuter des formulaires.
- AForm : classe abstraite définissant les règles communes (signature, exécution, restrictions de grade).
- Formulaires concrets :
  - ShrubberyCreationForm : crée un fichier ASCII d'arbre.
  - RobotomyRequestForm : robotomisation aléatoire 50%.
  - PresidentialPardonForm : pardon officiel.
- Intern : stagiaire capable de créer un formulaire selon un nom textuel.

AForm contient :

- Un nom (const)
- Un état signé ou non
- Un grade minimal pour signer
- Un grade minimal pour exécuter
- Une méthode virtuelle pure execute()

Chaque formulaire concret doit :

- Définir son constructeur avec target
- Définir execute()
- Proposer une méthode create() pour l'Intern

### 2. FONCTIONNEMENT DU INTERN

Intern::makeForm() prend deux chaînes : formName et target.

Le stagiaire :

1. Compare le nom demandé avec un tableau de noms disponibles.
2. Utilise un tableau parallèle de pointeurs de fonctions créatrices.
3. Affiche "Intern creates " en cas de succès.
4. Retourne un pointeur vers un AForm nouvellement créé.
5. En cas d'erreur, affiche :

Intern cannot create form "": unknown form name

puis retourne NULL.

C'est une implémentation propre, sans if/else en cascade.

### 3. INTERACTION BUREAUCRAT / FORMULAIRE

Un formulaire doit être :

1. Signé avant d'être exécuté
2. Exécuté par un bureaucrate dont le grade est assez élevé

La méthode checkExecutable() vérifie ces restrictions.

Les exceptions GradeTooLowException, GradeTooHighException et FormNotSignedException garantissent que les règles sont respectées.

### 4. ANALYSE COMPLETE DE LA SORTIE DU PROGRAMME

A — ShrubberyCreationForm

"Boss executes ShrubberyCreationForm"

Peon (grade 150) tente de signer → échoue.

Boss (grade 1) signe → succès.

Boss exécute → création du fichier "home\_shrubbery".

Message final : exécution réussie par Boss.

B — RobotomyRequestForm

Intern creates robotomy request

Boss signs RobotomyRequestForm

\* Drilling noises \*

The robotomy on Bender has failed...

Boss executes RobotomyRequestForm

Analyse :

- Le stagiaire reconnaît le formulaire.

- Boss signe avec succès (grade 1 < 72).
- L'exécution est aléatoire (50% réussite).
- Message final : Boss a exécuté.

#### C — PresidentialPardonForm

Intern creates presidential pardon

Boss signs PresidentialPardonForm

Arthur Dent has been pardoned by Zafod Beeblebrox.

Boss executes PresidentialPardonForm

Analyse :

- Le stagiaire crée correctement le formulaire.
- Boss signe (grade requis : 25).
- Exécution : message imposé par le sujet.

#### D — Formulaire inconnu

Intern cannot create form "coffee making": unknown form name

Analyse :

- Le stagiaire ne connaît que 3 noms.
- Message clair + retour NULL → comportement conforme au sujet.

### 5. CONTRE-EXEMPLES POUR LA SOUTENANCE

a) Exécuter un formulaire non signé :

→ FormNotSignedException

b) Exécuter avec un grade insuffisant :

→ GradeTooLowException

c) Demander un nom inconnu :

→ Retour NULL + message d'erreur explicite.

### 6. QUESTIONS FREQUENTES EN SOUTENANCE & RÉPONSES

Q : Pourquoi AForm est abstraite ?

R : Elle définit les règles mais pas l'action, qui dépend du formulaire concret.

Q : Pourquoi utiliser un tableau de pointeurs de fonctions ?

R : Pour éviter un gros bloc de if/else, comme demandé par le sujet.

Q : Pourquoi le Robotomy est aléatoire ?

R : C'est une exigence du sujet : "about 50% of the time".

Q : Comment éviter les fuites mémoire ?

R : Chaque new dans makeForm est libéré avec delete dans le main.

### 7. CONCLUSION

Ce projet illustre l'héritage, le polymorphisme, la conception propre d'une fabrique (Intern), la gestion d'exceptions et le contrôle d'accès hiérarchique. La sortie obtenue confirme le fonctionnement correct du système et le respect strict du sujet CPP05.